IREM de Clermont-Ferrand
Formation ISN
Algorithmique

20 octobre 2011
Malika More
malika.more@u-clermont1.fr

Tableaux et listes chaînées

1 Tableaux

Traiter des séries de données

Exemple (Un triangle $M_1M_2M_3$ est-il isocèle en M_1 ?).

- Soient Les points

$$M_1(x_1, y_1) \qquad M_2(x_2, y_2) \qquad M_3(x_3, y_3)$$

- On compare les longueurs

$$M_1M_2 \qquad \text{et} \qquad M_1M_3$$

- où

$$M_1M_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$M_1M_3 = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$

Algorithme 1 : TriangleIsocèle

début
  pour i **de** *1* **à** *3* **faire**
    Lire la valeur de $abs[i]$ `% série des abscisses %`
    Lire la valeur de $ord[i]$ `% série des ordonnées %`
  fin
  Donner à $d1$ la valeur $\sqrt{(abs[2] - abs[1])^2 + (ord[2] - ord[1])^2}$
  Donner à $d2$ la valeur $\sqrt{(abs[3] - abs[1])^2 + (ord[3] - ord[1])^2}$
  si $d1 = d2$ **alors**
    Afficher "C'est un triangle isocèle en M_1"
  sinon
    Afficher "Ce n'est pas un triangle isocèle en M_1"
  fin
fin

Notions élémentaires

6	3	7	2	3	5

Tableau

- Utilisé pour stocker plusieurs données de même type
- Type des données : simple ou élaborée
- Longueur du tableau : nombre de cases n
- Indices des cases : de 1 à n ou de 0 à $n-1$ ou autre
- Accès direct au contenu de la case d'indice i par $T[i]$
- Permet un traitement séquentiel par des boucles

Exemple d'utilisation

Exercice 1. Écrire un algorithme prend en entrée une valeur n et renvoie la moyenne de n nombres pris au hasard entre 1 et 100

Fonction `Moyenne`(n : *entier*)

variables locales : $i, somme$: entiers, $moyenne$: flottant, T : tableau de n entiers

début

  pour i **de** *1* **à** n **faire**

    Donner à $T[i]$ une valeur au hasard entre 1 et 100

  fin

  Donner à $somme$ la valeur 0

  pour i **de** *1* **à** n **faire**

    Donner à $somme$ la valeur $somme + T[i]$

  fin

  Donner à $moyenne$ la valeur $somme/n$

  retourner : $moyenne$

fin

Notions avancées

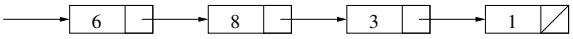
6	3	7	2	3	5

Tableau

- Tableau à plusieurs dimensions $\longrightarrow$ matrices etc.
- Si le nombre d'éléments peut varier en cours de calcul, on préfèrera souvent utiliser une liste (à suivre...)
- Dans les langages de programmation, les indices des cases commencent obligatoirement à une valeur fixée (en général 0 ou 1)
- En termes de complexité, le nombre d'éléments est en général une donnée plus pertinente que le type des éléments

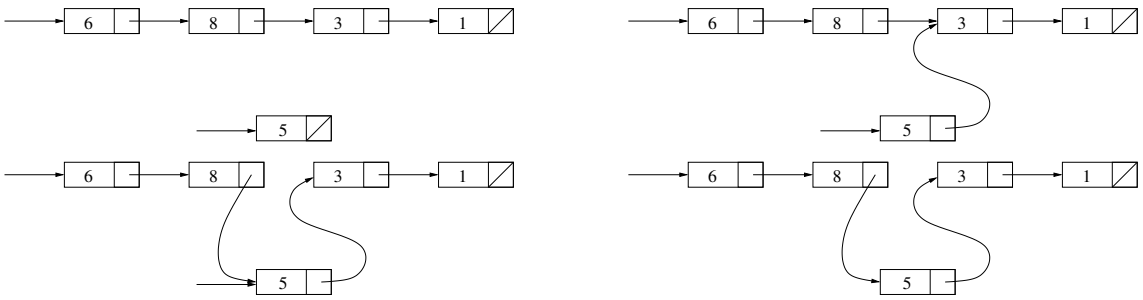
2 Listes chaînées

Listes chaînées



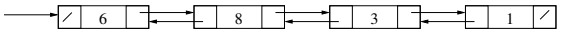
L'autre structure séquentielle

- Chaque cellule contient
  - ⋆ Une donnée
  - ⋆ L'adresse de la cellule suivante (un « pointeur »)
- Pas d'accès direct au contenu des cellules
  - ⋆ Accès à la première cellule via un pointeur
  - ⋆ Accès aux cellules suivantes par lecture séquentielle
- Structure dynamique
  - ⋆ Insérer/supprimer des cellules
  - ⋆ Longueur variable

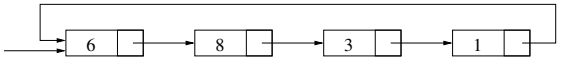


Compléments
- Données simples ou complexes (d'autres listes...)
- Boucles `tant que`
- En termes de complexité, la longueur maximale atteinte par la liste est en général la donnée pertinente
- Les pointeurs existent dans la plupart des langages
- Selon les langages, les primitives de gestion des listes sont transparentes pour l'utilisateur

Variantes
- Liste doublement chaînée



- Liste circulaire



Représentation des polynômes

Un polynôme : $2x^6 + 2x^3 + 5x + 4$

On dérive : $12x^5 + 6x^2 + 5$

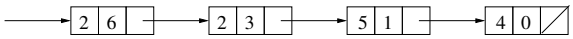
Un tableau :

4	5	0	2	0	0	2
0	1	2	3	4	5	6

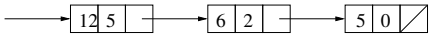
On dérive :

5	0	6	0	0	12	0
0	1	2	3	4	5	6

Une liste :



On dérive :

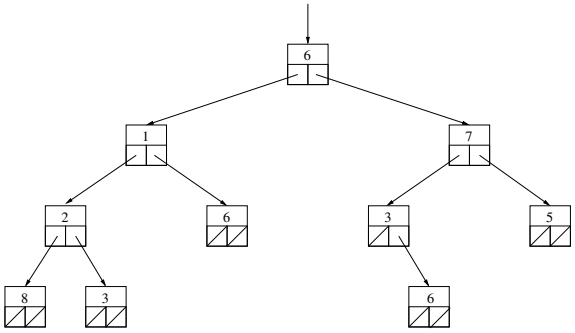


Fonction `Deriv`(P : *Monôme*)

début
  Donner à $Poly$ la valeur P
  si $Poly \neq$ `NULL` **alors**
    Donner à $Coef(Poly)$ la valeur $Coef(Poly) \times Deg(Poly)$
    si $Deg(Poly) \neq 0$ **alors**
      Donner à $Deg(Poly)$ la valeur $Deg(Poly) - 1$
      si $Suiv(Poly) \neq$ `NULL` **et** $Deg(Suiv(Poly)) = 0$ **alors**
        Donner à $Suiv(Poly)$ la valeur `NULL`
      fin
      `% Appel récursif : monôme suivant %`
      `Deriv`($Suiv(Poly)$)
    fin
  fin
fin

3 Arbres

Arbres binaires



Arbres généraux